Universidad Tecnológica de Pereira. ISSN 0122-1701

Ecosistema Big Data en un clúster de Raspberry Pi

Big Data ecosystem in a Raspberry Pi cluster

Sebastian Lopez Ramirez, Oscar Andrés Preciado Guevara
Ingeniería de Sistemas y Computación, Universidad Tecnológica de Pereira, Pereira, Colombia
Correo-e: sebastialopez@utp.edu.co, oapreciado@utp.edu.co

***Resumen*— Esta investigación mostrara un paso a paso de como instalar y configurar Hadoop en un clúster de raspberrys pi, describiendo y explicando desde los fundamentos de Big Data hasta todo el ecosistema de Apache y para que funciona cada tecnología. Además de recopilar información de algunas de las publicaciones mas relevantes relacionadas con Big Data.**

***Palabras clave*— Apache, Big Data, CAP, Ciencia de datos, Flume, Hadoop, HBase, Hive, NoSQL, Raspberry Pi, Raspbian, SQL, Sqoop, YARN, ZooKeeper.**

***Abstract*— This research shows a step by step of how to install and configure Hadoop in a cluster of raspberrys pi, describing and explaining from the basics of Big Data to the entire Apache ecosystem and how each technology works. In addition to collecting information from some of the most relevant publications related with Big Data.**

***Key Words* — Apache, Big Data, CAP, Data science, Flume, Hadoop, HBase, Hive, NoSQL, Raspberry Pi, Raspbian, SQL, Sqoop, YARN, ZooKeeper.**

I. INTRODUCCIÓN

Big Data es un concepto que hace referencia al almacenamiento de grandes cantidades de datos y a los procedimientos y conjuntos de herramientas usados para encontrar patrones repetitivos dentro de estos datos, siendo tan grande esta cantidad de datos que supera la capacidad del software convencional para ser capturados, administrados y procesados en tiempo real, la cual es una de las virtudes del Big Data. Es importante también resaltar que si bien un conjunto de datos es grande, no es Big Data, la clave está en el número de fuentes que se tiene para crear aquel dataset, es decir, entre más fuentes más complejo es el conjunto de datos y más difícil es de trabajar con técnicas de análisis convencionales.

Las fases por las cuales debe de pasar esta información para lograr encontrar patrones dentro de la misma empiezan en la recolección y el almacenamiento, después viene la búsqueda, distribución, análisis y por último la visualización clara de estos patrones y/o conclusiones. Esta tendencia ha ido creciendo en los últimos años debido a la necesidad de crear informes estadísticos y modelos predictivos que ayuden a la toma de decisiones en diferentes materias, tales como los análisis de negocios, los datos de enfermedades infecciosas, el espionaje y seguimiento a la población o hasta en la lucha contra el crimen organizado.

Para poder entender de forma clara el fenómeno de Big Data debemos primero definir qué tipos de datos son los que se tratan con este conjunto de técnicas, estos están divididos en tres grandes grupos: Datos estructurados, los cuales tienen un formato bien definido como por ejemplo, las bases de datos relacionales y las hojas de cálculo; después vienen los datos no estructurados, se almacenan de forma llave valor y o que pueden soportar almacenamiento de datos tales como documentos multimedia, documentos de texto o grafos; y por último se encuentran los datos semi estructurados, que contienen marcadores para separar sus elementos y poseen sus propios metadatos estructurados y emplean formatos como HTML, XML o JSON, ejemplo de ello es el correo electrónico.

Todos estos datos son generados cada segundo directa o indirectamente, cuyo origen puede provenir de un correo electrónico, estados en Facebook, resultado de transacciones de datos de facturación, llamadas telefónicas, transacciones entre cuentas bancarias, transacciones entre máquinas (M2M), o resultado de sensores o equipos médicos que generan información comúnmente no almacenada.

Después de recolectar todos estos datos muy seguramente tendremos una cantidad impresionante de tablas sin ninguna relación, aquí juega un papel importante las plataformas ETL (Extract, Transform and Load) las cuales tienen como propósito realizar conversiones de datos, limpieza de datos sucios, cambios de formato, etc., para cargar estos datos ya transformados en una base de datos específica, las cuales suelen ser NoSQL pues ofrecen un modelo de datos más flexible que el esquema entidad-relación y permite manipular grandes cantidades de información de manera más rápida que las bases de datos tradicionales (relacionales).

Cuando ya tenemos todos estos datos listos para su

Fecha de Recepción: 12 de septiembre de 2018

análisis podemos encontrar que existen diferentes técnicas. Algunas de las más utilizadas son:

- Asociación, la cual permite encontrar relaciones entre diferentes variables.
- Minería de datos, que tiene como objetivo encontrar comportamientos predictivos haciendo uso de técnicas que combinan métodos estadísticos y de machine learning con almacenamiento en bases de datos.
- Clustering, la cual es un tipo de minería de datos que divide grandes grupos de individuos en grupos más pequeños de los cuales no se conocían sus semejanzas antes del análisis.
- Análisis de texto, la cual permite extraer información clave de textos como e-mails, búsquedas web o contenidos.

Por último, después de realizar el análisis se deben de visualizar los resultados de una manera entendible, y tan amigable como sea posible mediante gráficos y tablas de resultados estadísticos, mapas o infografías que puedan soportar las conclusiones extraídas del modelo.

II. PLANTEAMIENTO DEL PROBLEMA

Si empezamos a buscar información acerca de big data, nos daremos cuenta que cada día se generan nuevos datos, y aunque algunos son fáciles de accesar, otros, igual de importantes, se pierden en ese mar de información. Según cifras del Ministerio de TIC, el 36% de las instituciones públicas no hace un inventario de los datos que genera, el 27% hace un inventario parcial y el 37% hace un inventario completo. El 66% de las entidades del Estado publica sus datos, mientras que el 34% no lo hace.

La importancia de estos datos y su buen manejo pueden influir la balanza de decisiones empresariales en manejo de clientes y proveedores, innovaciones, logística, procesos de producción. La toma de decisiones puede llevar a una empresa hasta la cima o a la quiebra, incluso las entidades gubernamentales necesitan realizar constantemente análisis de sectores, tasas de desempleo, reducción de criminalidad, o generar alertas tempranas en el área de la salud para prevenir o controlar epidemias donde existen aplicaciones tales como google flu trends que permite relacionar las personas que buscan sobre la gripe y su lugar de residencia.

III. OBJETIVO GENERAL

- Realizar una guía con el procedimiento básico necesario para poder instalar y administrar un entorno Big Data en un clúster de Raspberrys Pi.

IV. OBJETIVOS ESPECIFICOS

- Enumerar de forma clara cada componente de un ecosistema Big Data y sus términos más importantes.
- Realizar una guía sobre como instalar un SO en una Raspberry Pi.
- Hacer un manual que ilustre el proceso de instalación y configuración de Hadoop en un clúster de Raspberrys Pi.
- Hacer un instructivo donde se registren los procedimientos necesarios que nos permitan el correcto funcionamiento del clúster.

V. JUSTIFICACION DEL PROYECTO

En las bases del Plan Nacional de Desarrollo 2014-2018 se estipuló que "El DNP liderará la estrategia de Big Data del Estado, la cual definirá la política de innovación basada en datos. Esta permitirá el procesamiento y análisis de los datos, como recurso de infraestructura, para el desarrollo de nuevos conocimientos, la creación de valor, el surgimiento de nuevos productos, servicios, procesos y mercados…". Basado en lo anterior podemos ver el compromiso que tiene el estado con esta nueva tecnología, ya que están viendo la necesidad de implementarla, porque el volumen de los datos que manejan es demasiado grande. Un ejemplo claro de la dimensión de los datos que se recopilan son los del DNP. Con esto nos permitiremos dejar de revisar los errores del pasado y dar una mirada hacia el futuro, con un análisis mucho más predictivo ya que este se hace casi en tiempo real.

Este proyecto tiene además mucha validez no solo en la Ingeniería de sistemas y computación sino en toda la Universidad Tecnológica de Pereira y por qué no en el departamento de Risaralda, ya que no solo se pondrán en práctica conocimientos adquiridos durante la carrera, sino que será algo realmente innovador, un tema sobre el cual casi nadie se ha atrevido a hacer una implementación a pesar de existir la teoría y sobre todo logrando un impacto a la maestría de Sistemas, ya que a futuro se podría implementar un laboratorio de datos en la Universidad siendo uno de los primeros, si no es el primero en el país.

VI. ESTADO DEL ARTE

Título: Big data: ¿Solución o problema?
Autor (es): Pulido Cañabate, Estrella
Palabras clave: Big Data; Solución de problemas; Generación de problemas; Informática
Resumen: El documento nos da una introducción acerca de la situación actual con respecto a la generación de una cantidad de datos que pueden pasar desapercibidos pero tienen un alto valor. También nos habla de que a pesar de que existen muchas fuentes, de las más importantes se encuentran en transacciones bancarias, compras por internet y maquinas industriales como el acelerador de partículas del CERN. Nos explica que el big data ha tenido potencia gracias a que hoy en día los costos de

almacenamiento se han vuelto bastante económicos comparados con hace unos 20 o 30 años, y la aparición de software inteligente que permite el análisis de esos datos, y los nuevos tipos de datos como los generados por redes sociales. Sin embargo así como hay cosas positivas como las mejoras a la educación y la salud, también debemos darnos cuenta que hay un déficit de personas capacitadas para el manejo de este campo, además de que se manejan datos muy sensibles que podrían hacer que algunas personas sean vulneradas en su privacidad, o abusos intencionados.
El aporte que este documento genera a nuestro proyecto de grado, es uno de los más valiosos ya que no solo nos da unas bases de conocimiento para entender el porqué del big data hoy en día, sino que nos muestra las dos caras de la moneda. Si bien vemos el big data como una solución a múltiples problemas, debemos ser muy cuidadosos al momento de usar esta tecnología ya que si esta información es mal manipulada podríamos vernos en un escenario totalmente diferente al que aspiramos se vea. Todo va a estar relacionado con el buen manejo y la responsabilidad que le demos a la información manipulada, además de la protección
de dichos datos.

Título: Estrategias, herramientas y métricas de posicionamiento de Big Data en laboratorios de investigación universitarios en redes sociales
Autor (es): Herrera de los Ríos, Ismael
Palabras clave: Datos masivos; Apache Hadoop (Servidor Web); Redes sociales; Telecomunicaciones
Resumen: Este documento nos orienta sobre cómo el big data se puede ver reflejado de diferentes maneras en diferentes empresas y soluciones como por ejemplo el ámbito empresarial, el ámbito deportivo, como ciencia de datos e incluso en las compañías aseguradoras. También nos muestra cómo se realiza el diseño de una arquitectura de big data con componentes como HDFS, bases de datos no relacionales, MAPREDUCE, algunas herramientas de visualización de datos como ease.ly o piktochart, y descripción detallada de una arquitectura hadoop con todos sus componentes. Luego esto se ve reflejado en la solución del tema planteado que es el funcionamiento de big data en redes sociales el cual nos muestra la cantidad de datos que a veces pasamos por alto pero que siempre se están recolectando día a día por millones de usuarios de estas redes, tomando Twitter como ejemplo en dicha universidad.
El aporte de este documento a este proyecto de grado no solo es teórico-básico, gracias a la excelente explicación de cada módulo utilizado en el diseño y la arquitectura, sino que también nos orienta sobre qué herramientas nos pueden facilitar el montaje de esta, además de una pequeña guía de instalación de hadoop, y estadísticas que podemos tomar de referencia para que este trabajo sea
completo y lo mejor detallado posible.

Título: Técnicas Big Data para el análisis de datos de tráfico de red sobre Hadoop
Autor (es): García-Valcárcel Sen, Rubén
Palabras clave: Big Data; Sistemas distribuidos; Informática
Resumen: Este documento nos describe como Apache Hadoop es una opción clara al hacer procesos distribuidos ya que hoy en día con un servidor normal y a pesar de su buen rendimiento, no parece ser suficiente para la cantidad de datos que se está manejando hoy en día. Por ello es que se recurre a esta herramienta, porque nos da la facilidad de capturar, almacenar, procesar y analizar todo ese flujo de datos que comúnmente no son tomados en cuenta, y además se añade una interfaz interactiva para la visualización de los datos utilizando Apache Hive, y ya que este está basado en SQL, simplifica las consultas necesarias. Por último se realizan una serie de pruebas que nos muestran que esta herramienta puede realizar cálculos de una tasa de procesamiento superior a 7 GBPS, que es superior al de las herramientas estándar que están actualmente disponibles.
El aporte que nos genera este documento es muy significativo ya que nos muestra como está trabajando el sistema de manera funcional, no teórica, nos explica qué herramientas se usaron para la captura del tráfico de datos, una descripción detallada de la red que se usó para el análisis y un ejemplo ilustrado de una posible
arquitectura en la cual nos podemos basar para el desarrollo de nuestro proyecto de grado. Por último también nos muestra unos ejemplos tipo experimentos realizados con la herramienta que nos da bases para poder realizar nuestros propios ensayos.

Big data, Big bang?
Autor: Jacques Bughin
Palabras Clave: Information systems, Big data, Data analytics, Competitive performance, Organization assets
Resumen:
Utilizando una muestra aleatoria compuesta por cientos de empresas en todo el mundo, se probó el impacto en el rendimiento de la empresa de invertir en proyectos de big data enfocados en tres áreas principales de negocios (el saber, la interfaz del cliente, la cadena de suministro de la empresa y la competencia). La prueba de rendimiento se basa en la llamada función de producción trans-logarítmica, lo que permite una prueba más directa de la complementariedad entre el capital de big data y las inversiones de trabajo de big data; Además, utilizaron una corrección de Heckman para ajustar el hecho de que las empresas que invierten en big data generalmente son más productivas que sus pares.
Se confirman y amplían los resultados iniciales de un impacto en la productividad del big data. Se encuentra que para el promedio de la muestra, las empresas más productivas también adoptan más rápido los grandes datos que sus pares de la industria (esto explica el 2,5% de la diferencia de productividad). Las grandes inversiones de datos en la arquitectura laboral y de TI son complementos, con un efecto de crecimiento de la productividad total de alrededor del 5,9%. Los grandes proyectos de datos dirigidos a los clientes y los dominios de inteligencia competitiva ofrecen un rendimiento

ligeramente mayor que los proyectos de big data destinados a las mejoras de la cadena de suministro.

Aporte a la tesis:

Este artículo amplía el trabajo reciente sobre los efectos del big data en el desempeño corporativo. Su principal innovación es desarrollar un enfoque más general del desempeño corporativo y de la función de producción, que nos permita responder preguntas más directas, como la (importancia de) la complementariedad del capital de grandes datos y la mano de obra.

Después de ajustar las empresas de mayor rendimiento como primeros usuarios de big data mediante un procedimiento de Heckman (1979), encontramos que el principal impacto en el rendimiento del big data reside en la estrecha complementariedad entre la inversión en TI de big data y las habilidades laborales. El efecto de rendimiento también es ligeramente mayor para los dominios de aplicación, como la inteligencia empresarial y los dominios de interfaz del cliente.

Las vías de investigación son claras. La primera avenida, irónicamente, se relaciona con mejores datos en sí mismos. Solo recopilamos información de corte transversal y nos gustaría ver cómo los grandes datos evolucionan a lo largo de los años, incluido el tiempo de desarrollo de la productividad. La segunda vía es investigar las interacciones más grandes entre las formas tradicionales de capital y trabajo y big data. Se supone además independencia relativa entre estos tipos de entradas, ya que de lo contrario, una estimación de regresión típica se vuelve rápidamente demasiado compleja.

A survey of open source tools for machine learning with big data in the Hadoop ecosystem

Autores: Sara Landset, Taghi M. Khoshgoftaar, Aaron N. Richter y Tawfiq Hasanin

Palabras Clave: Machine learning, Big data, Hadoop, Mahout, Mllib, SAMOA, H2O, Spark, Flink, Storm

Resumen:

Con una cantidad cada vez mayor de opciones, la tarea de seleccionar herramientas de aprendizaje automático para Big Data puede ser difícil. Las herramientas disponibles tienen ventajas e inconvenientes, y muchas tienen usos superpuestos. Los datos mundiales están creciendo rápidamente, y las herramientas tradicionales para el aprendizaje automático se están volviendo insuficientes a medida que avanzamos hacia el procesamiento distribuido y en tiempo real. Para evaluar las herramientas, uno debe tener una comprensión profunda de lo que debe buscar. Con ese fin, este documento proporciona una lista de criterios para hacer selecciones junto con un análisis de las ventajas y desventajas de cada uno. Hacemos esto comenzando desde el principio, y mirando qué significa exactamente el término "big data". A partir de ahí, pasamos al ecosistema de Hadoop para ver muchos de los proyectos que forman parte de una arquitectura típica de aprendizaje automático y una comprensión de cómo encajarían todas las cosas. Se discuten las ventajas y desventajas de tres paradigmas de procesamiento diferentes junto con una comparación de los motores que los implementan, incluyendo MapReduce, Spark, Flink, Storm y H2O. A continuación, analizamos las bibliotecas y los marcos de aprendizaje automático, incluidos Mahout, MLlib, SAMOA, y los evaluamos según criterios como la escalabilidad, la facilidad de uso y la extensibilidad. No existe un solo conjunto de herramientas que realmente incorpore una solución única para todos, por lo que este documento tiene como objetivo ayudar a que las decisiones sean más fluidas al proporcionar la mayor cantidad de información posible y cuantificar cuáles serán las ventajas y desventajas. Además, a lo largo de este documento, revisamos investigaciones recientes en el campo utilizando estas herramientas y hablamos sobre las posibles direcciones futuras para el aprendizaje basado en herramientas.

Aporte a la tesis:

Las tendencias actuales en tecnología, como la mayor adopción de computadoras portátiles y otros dispositivos de Internet de las cosas (IoT), permiten un acceso sin precedentes a cantidades masivas de datos heterogéneos. El aprendizaje eficiente a partir de estos datos a menudo requiere arquitecturas complejas que utilizan una combinación de herramientas y técnicas para la recopilación, almacenamiento, procesamiento y análisis. Armar una arquitectura de este tipo sería extremadamente difícil, incluso si hubiera opciones limitadas para elegir. La comunidad de ciencia de datos de fuente abierta es prolífica, lo que da como resultado muchas opciones y muchas más decisiones por tomar.

Como los conceptos de aprendizaje automático se adoptan cada vez más en los entornos de investigación y producción, la necesidad de herramientas para facilitar las tareas de aprendizaje es cada vez más importante. Muchas de estas herramientas son muy recientes y se necesita más investigación para comparar y evaluar adecuadamente todas las diferentes opciones. Se hablo sobre el ecosistema de Hadoop y una serie de herramientas que forman parte de él para proporcionar un contexto de cómo el aprendizaje automático se ajusta a un entorno de análisis. Se describieron tres enfoques principales del procesamiento (lote, lote iterativo y transmisión en tiempo real) y se presentaron y compararon los proyectos que los usaban. Además, se presentó una lista de criterios para la evaluación y selección de marcos de aprendizaje automático junto con una mirada en profundidad a proyectos ampliamente utilizados y prometedores, con una discusión de sus ventajas y desventajas.

Se eligieron proyectos en el ecosistema de Hadoop por varias razones. Primero, se encuentran entre los más innovadores que se haya visto. Además, existen pocos servicios de extremo a extremo para el aprendizaje automático y los proyectos de Hadoop tienden a diseñarse con la intención de conectarse con los que ya existen en este grupo. Finalmente, los proyectos de Apache tienden a atraer a un gran número de usuarios activos que son útiles cuando surgen problemas.

La elección de las herramientas dependerá en gran medida de las aplicaciones para las que se utilizan y de las preferencias del usuario. Por ejemplo, Mahout, MLlib, Flink-ML y Oryx incluyen opciones de recomendaciones, por lo que si la aplicación deseada es un sitio de comercio electrónico o una red social, es posible que desee elegir características como sugerencias de artículos o de usuarios. Las redes sociales o los

datos de IoT pueden requerir resultados en tiempo real, lo que requiere el uso de Storm o Flink junto con sus bibliotecas ML asociadas. Otros dominios, como el cuidado de la salud, a menudo producen conjuntos de datos dispares que pueden requerir una mezcla de procesamiento por lotes y transmisión, en cuyo caso Flink, Oryx o Spark serían la mejor opción.
En este documento, se examinaron cinco plataformas de procesamiento. MapReduce, una vez que el estándar de facto para proyectos de big data, se está quedando anticuado en la comunidad de aprendizaje automático y no se recomienda para la mayoría de las aplicaciones debido a su lentitud y falta de soporte para algoritmos iterativos. Spark es visto por muchos como un sucesor natural. Se basa en MapReduce por lo que la transición no es difícil, pero ofrece soporte para tareas iterativas y es capaz de soportar todas las bibliotecas de aprendizaje automático que MapReduce hace más otras. Sin embargo, si las soluciones en tiempo real son importantes, uno puede considerar Storm o Flink en su lugar, ya que ofrecen el procesamiento de flujo verdadero mientras que el uso de microbatch de Spark puede tener un pequeño retraso asociado a la recepción de resultados. Flink ofrece lo mejor de ambos mundos en este sentido, con una combinación de procesamiento por lotes y flujo real, pero es un proyecto muy joven y necesita más investigación sobre su viabilidad. Además, actualmente no admite casi tantas soluciones de aprendizaje automático como otras plataformas. H2O es el único sistema de extremo a extremo que se analiza en este documento y ofrece dos funciones que no están presentes en otros sistemas, que son una interfaz gráfica de usuario y soporte para el aprendizaje profundo. Además, admite tantas o más herramientas de aprendizaje automático que cualquiera de los otros motores que estudiamos. Al igual que Flink, hay muy poca investigación sobre el H2O, por lo que se necesita más para una evaluación adecuada.

Big data analytics: a survey
Autores: Chun-Wei Tsai, Chin-Feng Lai, Han-Chieh Chao y Athanasios V. Vasilakos
Palabras Clave: Big data, data analytics, data mining
Resumen:
La era del big data está llegando. Pero el análisis de datos tradicional puede no ser capaz de manejar cantidades tan grandes de datos. La pregunta que surge ahora es cómo desarrollar una plataforma de alto rendimiento para analizar de manera eficiente grandes volúmenes de datos y cómo diseñar un algoritmo de minería adecuado para encontrar las cosas útiles a partir de grandes volúmenes de datos.
A medida que la tecnología de la información se extiende rápidamente, la mayoría de los datos nacieron tanto digitales como intercambiados hoy en Internet. Según la estimación de Lyman y Varian, los nuevos datos almacenados en dispositivos de medios digitales ya han sido más del 92% en 2002, mientras que el tamaño de estos nuevos datos también fue más de cinco exabytes. De hecho, los problemas de analizar los datos a gran escala no se produjeron repentinamente, pero han estado allí durante varios años porque la creación de datos suele ser mucho más fácil que encontrar datos útiles a partir de los datos. A pesar de que los sistemas informáticos de hoy en día son mucho más rápidos que los de la década de 1930, los datos a gran escala son una tensión para analizar por las computadoras que tenemos hoy en día.
En respuesta a los problemas de análisis de datos a gran escala, bastantes métodos eficientes, como el muestreo, la condensación de datos, los enfoques basados en densidad, los enfoques basados en cuadrículas, dividir y conquistar, aprendizaje incremental y computación distribuida, han sido presentado. Por supuesto, estos métodos se usan constantemente para mejorar el rendimiento de los operadores del proceso de análisis de datos.
Aunque los avances de los sistemas informáticos y las tecnologías de Internet han sido testigos del desarrollo del hardware informático siguiendo la ley de Moore durante varias décadas, los problemas de manejo de los datos a gran escala aún existen cuando estamos entrando en la era del big data. Es por eso que Fisher señaló que los datos masivos significan que la mayoría de los sistemas o métodos de información actuales no pueden manejar y procesar los datos porque los datos en la era del big data no solo serán demasiado grandes para ser cargados en una sola máquina, también implica que la mayoría de los métodos tradicionales de minería de datos o análisis de datos desarrollados para un proceso centralizado de análisis de datos pueden no aplicarse directamente a big data. Además de los problemas del tamaño de los datos, Laney presentó una definición bien conocida (también llamada las 3V) para explicar qué es big data: volumen, velocidad y variedad. La definición de las 3V implica que el tamaño de los datos es grande, los datos se crearán rápidamente y los datos se incluirán en múltiples tipos y se capturarán de diferentes fuentes, respectivamente. Estudios posteriores señalaron que la definición de 3V es insuficiente para explicar el big data que enfrentamos ahora. Por lo tanto, la veracidad, la validez, el valor, la variabilidad, el lugar, el vocabulario y la vaguedad se agregaron para hacer una explicación complementaria de los grandes datos.
Aporte a la tesis:
Para el tiempo de cálculo, no hay duda de que la computación paralela es una de las tendencias futuras importantes para que el análisis de datos funcione para big data, y en consecuencia las tecnologías de computación en la nube, Hadoop y map-reduce jugarán los roles importantes para el análisis de big data. Dado que los problemas de manejo y análisis de datos de entrada complejos y de gran escala siempre existen en el análisis de datos, se presentaron varios métodos de análisis eficientes para acelerar el tiempo de cálculo o reducir el costo de memoria para el proceso KDD. El estudio muestra que los conceptos matemáticos básicos (es decir, la desigualdad de triángulo) se pueden utilizar para reducir el costo de cálculo de un algoritmo de agrupamiento. Otro estudio muestra que las nuevas tecnologías (es decir, la computación distribuida por GPU) también se pueden utilizar para reducir el tiempo de cálculo del método de análisis de datos. Además de los conocidos métodos mejorados para estos métodos de análisis (p. Ej., Desigualdad triangular o computación distribuida), una gran proporción de estudios diseñaron sus métodos eficientes

basados en las características de los algoritmos de minería o el problema en sí. Este tipo de métodos mejorados generalmente se diseñó para resolver el inconveniente de los algoritmos de minería o utilizar diferentes formas de resolver el problema de la minería.
Hoy en día, los datos que deben analizarse no son solo grandes, sino que están compuestos por varios tipos de datos e incluso incluyen datos de transmisión. Dado que Big Data tiene las características únicas de "masivo, de alta dimensión, heterogéneo, complejo, no estructurado, incompleto, ruidoso y erróneo", que puede cambiar los enfoques estadísticos y de análisis de datos. Aunque parece que el big data nos permite recopilar más datos para encontrar más información útil, la verdad es que más datos no necesariamente significan más información útil. Puede contener datos más ambiguos o anormales. Por ejemplo, un usuario puede tener múltiples cuentas, o una cuenta puede ser utilizada por múltiples usuarios, lo que puede degradar la precisión de los resultados de la minería. Por lo tanto, surgen varios problemas nuevos para el análisis de datos, como la privacidad, la seguridad, el almacenamiento, la tolerancia a errores y la calidad de los datos.

A survey on platforms for big data analytics
Autores: Dilpreet Singh y Chandan K Reddy
Palabras Clave: Big data, MapReduce, graphics processing units, scalability, big data analytics, big data platforms, k-means clustering, real-time processing
Resumen:
Esta es una era de Big Data. Big Data está impulsando cambios radicales en las plataformas de análisis de datos tradicionales. Para realizar cualquier clase de análisis sobre datos tan voluminosos y complejos, la ampliación de las plataformas de hardware se vuelve inminente y la elección de las plataformas adecuadas de hardware / software se convierte en una decisión crucial si los requisitos del usuario deben cumplirse en un período de tiempo razonable. Los investigadores han estado trabajando en la creación de nuevas técnicas de análisis de datos para Big Data más que nunca, lo que ha llevado al desarrollo continuo de muchos algoritmos y plataformas diferentes.
Hay varias plataformas de big data disponibles con diferentes características y elegir la plataforma adecuada requiere un conocimiento profundo de las capacidades de todas estas plataformas. Especialmente, la capacidad de la plataforma para adaptarse a las crecientes demandas de procesamiento de datos juega un papel fundamental a la hora de decidir si es apropiado construir soluciones basadas en análisis en una plataforma particular. Con este fin este paper proporciona un análisis en profundidad de las diferentes plataformas disponibles para realizar análisis de big data. Se examinan las diferentes plataformas de hardware disponibles para Big Data Analytics y evalúan las ventajas e inconvenientes de cada una de estas plataformas en base a diversas medidas como escalabilidad, velocidad de E / S de datos, tolerancia a fallas, procesamiento en tiempo real, tamaño de datos compatible y tareas iterativas. apoyo. Además del hardware, una descripción detallada de los marcos de software usados dentro de cada una de estas plataformas también se analiza junto con sus puntos fuertes y desventajas. Algunas de las características críticas que se describen aquí pueden ayudar a tomar una decisión informada sobre la elección correcta de las plataformas en función de sus necesidades de cómputo. Usando una tabla de clasificación de estrellas, también se discute una comparación cualitativa rigurosa entre diferentes plataformas para cada una de las seis características que son críticas para los algoritmos de análisis de big data. Para proporcionar más información sobre la efectividad de cada plataforma en el contexto del análisis de big data, los detalles específicos del nivel de implementación del algoritmo de clustering de k-means ampliamente utilizado en varias plataformas también se describen en el pseudocódigo de la forma.
Aporte a la tesis:
Este documento examina varias plataformas de procesamiento de datos que están actualmente disponibles y analiza las ventajas y desventajas de cada una de ellas. También se proporcionan varios detalles sobre cada una de estas plataformas de hardware junto con algunos de los marcos de software populares, como Hadoop y Spark. También se ha realizado una comparación minuciosa entre diferentes plataformas basada en algunas de las características importantes (como la escalabilidad y el procesamiento en tiempo real) a través de clasificaciones basadas en estrellas. El algoritmo de agrupamiento k-means ampliamente utilizado fue elegido como un estudio de caso para demostrar las fortalezas y debilidades de diferentes plataformas. Algunas de las características importantes del algoritmo k-means, como su naturaleza iterativa, los cálculos de cálculo intensivo y la agregación de resultados locales en un entorno paralelo, lo convierten en una opción ideal para comprender mejor las diversas plataformas de big data. Cabe señalar que muchos de los algoritmos analíticos también comparten estas características.
La decisión de elegir una plataforma particular para una determinada aplicación generalmente depende de los siguientes factores importantes: tamaño de los datos, velocidad o optimización del rendimiento y desarrollo del modelo. Ahora proporcionaremos más detalles sobre cada uno de estos factores.
Tamaño de los datos: El tamaño de los datos que se están considerando para el procesamiento es probablemente el factor más importante. Si los datos pueden caber en la memoria del sistema, generalmente no se requieren clústeres y la información completa se puede procesar en una sola máquina. Las plataformas tales como GPU, CPU multinúcleo, etc. se pueden usar para acelerar el procesamiento de datos en este caso. Si los datos no encajan en la memoria del sistema, hay que mirar otras opciones de clúster como Hadoop, Spark, etc. De nuevo, los clústeres Hadoop y Spark pueden manejar una gran cantidad de datos, pero Hadoop tiene herramientas y marcos bien desarrollados, aunque es más lento para tareas iterativas. El usuario debe decidir si necesita usar herramientas estándar disponibles para Hadoop o si desea optimizar el rendimiento del clúster, en cuyo caso Spark es más apropiado.

Velocidad u optimización del rendimiento: Aquí, la velocidad se refiere a la capacidad de la plataforma para procesar datos en tiempo real, mientras que el rendimiento se refiere a la cantidad de datos que el sistema es capaz de manejar y procesar simultáneamente. Los usuarios deberán tener claro si el objetivo es optimizar el sistema en cuanto a velocidad o rendimiento. Si uno necesita procesar una gran cantidad de datos y no tiene restricciones estrictas en el tiempo de procesamiento, entonces uno puede buscar sistemas que puedan escalar para procesar grandes cantidades de datos como Hadoop, redes punto a punto, etc. las plataformas pueden manejar datos a gran escala, pero generalmente toman más tiempo para entregar los resultados. Por otro lado, si uno necesita optimizar el sistema para la velocidad en lugar del tamaño de los datos, entonces deben considerar los sistemas que son más capaces de procesamiento en tiempo real, como GPU, FPGA, etc. Estas plataformas son capaces de procesar los datos en tiempo real, pero el tamaño de los datos admitidos es bastante limitado.

Entrenamiento / Aplicación de un modelo: En el análisis de datos, la capacitación del modelo generalmente se realiza fuera de línea y, por lo general, lleva mucho tiempo. Un modelo se aplica generalmente en un entorno en línea donde el usuario espera los resultados dentro de un corto período de tiempo (casi instantáneamente). Esto crea una gran necesidad de investigar diferentes plataformas para capacitar y aplicar un modelo dependiendo de la aplicación del usuario final. Por lo general, durante el proceso de capacitación, el usuario debe manejar una gran cantidad de datos de capacitación y dado que la capacitación se realiza fuera de línea, el tiempo de procesamiento no es crítico. Esto hace que las plataformas de escala horizontal sean adecuadas para entrenar un modelo. Cuando se aplica un modelo, se esperan resultados en tiempo real y, por lo general, se prefieren las plataformas de escala vertical. Las GPU son preferibles a otras plataformas cuando el usuario tiene restricciones estrictas en tiempo real y los datos pueden encajar en la memoria del sistema. Por otro lado, los clústeres de HPC pueden manejar más datos en comparación con las GPU, pero no son adecuados para el procesamiento en tiempo real en comparación con las GPU.

Implicaciones prácticas: Este trabajo puede proporcionar información útil para una amplia variedad de aplicaciones prácticas. Muchos usuarios e investigadores que trabajan en diferentes áreas de Big Data, desde sistemas en tiempo real hasta aplicaciones de procesamiento de datos a gran escala, pueden evaluar las ventajas y desventajas de las plataformas que utilizan actualmente. Esta encuesta será de gran ayuda para que comprendan si pueden mejorar los sistemas existentes eligiendo la plataforma adecuada. Las fortalezas y debilidades de cada una de las plataformas se destacan claramente en este documento. Más específicamente, para las aplicaciones de big data, a menudo hay una compensación entre los requisitos de análisis en tiempo real y la escalabilidad de los datos que se procesan. Para los primeros, las GPU serán una opción óptima y para las últimas plataformas de escalado horizontal, como Hadoop y Spark, son opciones óptimas. Por ejemplo, en algunos de los problemas de recomendación relacionados con millones de usuarios, es extremadamente importante contar con una plataforma escalable que pueda manejar grandes cantidades de procesamiento de datos y no es absolutamente necesario tener los resultados en tiempo real. En las aplicaciones de búsqueda web, debido a los miles de millones de páginas web disponibles, el proceso de indexación de la página web requerirá una plataforma altamente escalable que pueda indexar con precisión las páginas web fuera de línea. Sin embargo, para el componente en línea real donde el usuario ingresa la consulta y se espera que los resultados se obtengan en tiempo real, las plataformas verticales de escalado podrían ser más adecuadas. Por lo tanto, dependiendo de la aplicación, uno probablemente puede definir las necesidades más críticas y luego elegir el tipo correcto de plataforma en consecuencia.

An efficient strategy for the collection and storage of large volumes of data for computation

Autores: Uthayanath Suthakar, Luca Magnoni, David Ryan Smith, Akram Khan y Julia Andreeva

Palabras Clave: Big Data, Data pipeline, Hadoop, Apache, Flume, MapReduce, HDFS

Resumen:

El campo de la ciencia de datos se ha convertido en un tema ampliamente discutido en los últimos años debido a una explosión de datos, especialmente con experimentos científicos como los que forman parte del Gran Colisionador de Hadrones (LHC) en el CERN y empresas comerciales interesadas en mejorar su competitividad aprendiendo acerca de sus clientes para proporcionar productos y servicios a medida, aumentando drásticamente el uso de dispositivos de sensores. Las técnicas tradicionales de recopilar (por ejemplo, un framework de Python ligero), almacenar (por ejemplo, Oracle) y analizar (por ejemplo, PL / SQL) datos ya no son óptimos con la abrumadora cantidad de datos que se están generando. El desafío de manejar grandes volúmenes de datos ha sido asumido por muchas compañías, particularmente aquellas en el dominio de Internet, lo que lleva a un cambio total de paradigma en los métodos de archivo, procesamiento y visualización de datos. Han aparecido varias tecnologías nuevas, cada una dirigida a aspectos específicos del procesamiento de datos distribuidos a gran escala. Todas estas tecnologías, como los sistemas de computación por lotes (por ejemplo, Hadoop) y las bases de datos no estructuradas (por ejemplo, MongoDB), pueden manejar volúmenes de datos muy grandes con poco costo financiero. Por lo tanto, se hace necesario tener una buena comprensión de las tecnologías actualmente disponibles para desarrollar un marco que pueda respaldar la recopilación, el almacenamiento y el análisis de datos de forma eficiente.

En los últimos años, se ha producido y almacenado una cantidad cada vez mayor de datos, que se conoce como Big Data. Las redes sociales, Internet de cosas, experimentos científicos y servicios comerciales juegan un papel importante en la generación de una gran cantidad de datos. Tres factores principales son importantes en Big Data; Volumen, velocidad y variedad. Uno debe considerar los tres factores al diseñar una plataforma para soportar Big Data. El acelerador de partículas Large Hadron Collider (LHC) en el CERN consiste en una serie

de experimentos de uso intensivo de datos, que se estima que producen un volumen de aproximadamente 30 PB de datos al año. La velocidad de estos datos que se propagan será extremadamente rápida. Los métodos tradicionales de recopilación, almacenamiento y análisis de datos se han vuelto insuficientes para gestionar el volumen de datos en rápido crecimiento. Por lo tanto, es esencial tener una estrategia eficiente para capturar estos datos a medida que se producen. En este documento, se exploran una serie de modelos para comprender cuál debe ser el mejor enfoque para recopilar y almacenar Big Data para análisis. Se presenta una evaluación del rendimiento de los ciclos completos de ejecución de estos enfoques sobre el monitoreo de la infraestructura de la Grilla de computación del LHC a nivel mundial (WLCG) para recopilar, almacenar y analizar datos. Además, los modelos discutidos se aplican a una solución de software impulsada por la comunidad, Apache Flume, para mostrar cómo pueden integrarse sin problemas.

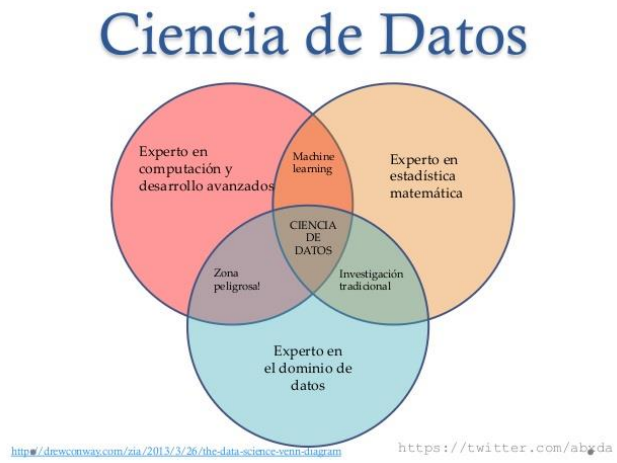
Aporte a la tesis:

Los enfoques propuestos para recopilar y almacenar Big Data para análisis presentados en este documento muestran la importancia de seleccionar el modelo correcto para un rendimiento eficiente y una migración de tecnología. Del estudio se desprende que mantener la lógica principal en una ubicación centralizada simplificará la migración tecnológica y arquitectónica. Los resultados de la prueba de rendimiento muestran que eliminar cualquier transformación en el nivel de ingestión de datos y moverlo al trabajo analítico es beneficioso ya que se reduce el tiempo total del proceso, los datos brutos sin templar se mantienen en el nivel de almacenamiento para la tolerancia a fallas y la transformación requerida puede hacerse cuando y como se requiera usando un framework como MapReduce. Los resultados presentados muestran que este enfoque propuesto superó el enfoque empleado en el WLCG y, después de este trabajo, el nuevo enfoque ha sido adoptado por el WLCG y se ha utilizado para recopilar, almacenar y analizar metadatos en el CERN desde abril de 2015. Este enfoque se puede aplicar fácilmente a otros casos de uso (por ejemplo, en empresas comerciales para recopilar conjuntos de datos de interés del cliente) y no se limita a aplicaciones científicas. El trabajo futuro incluirá observar cómo funcionará la cadena de datos en el nuevo enfoque si el marco de MapReduce fuera reemplazado por el ecosistema Spark que admite el procesamiento en memoria.

VII. MARCO TEORICO

Ciencia de datos

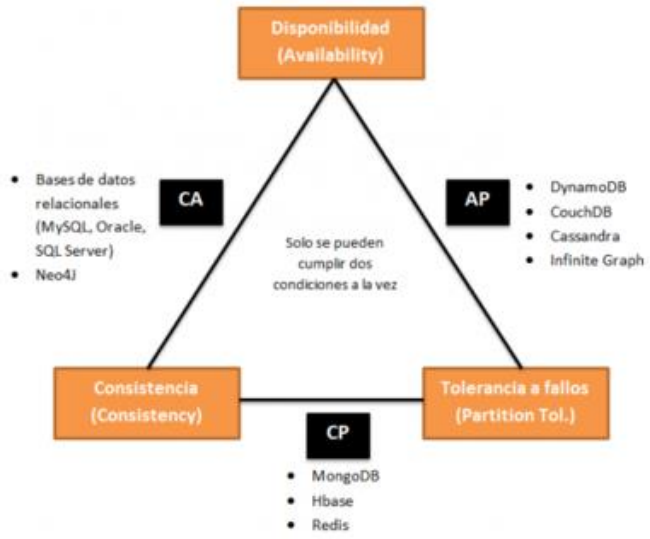
Se puede entender como ciencia de datos al conjunto de métodos, técnicas o sistemas que involucran la extracción de los datos ya sean estructurados o no estructurados, como lo son el procesamiento de señales, modelos probabilísticos, machine learning, aprendizaje estadístico, programación, ingeniería de datos, reconocimiento de patrones, visualización, modelización de la incertidumbre, data warehousing, y computación de altas prestaciones para su posterior análisis con diferentes herramientas como bases de datos SQL, big data, noSQL, lenguajes de programación como R, Python, programación distribuida, hadoop… Y todo esto con el fin de ayudar a gobiernos y compañías quienes están interesados y le han dado un nuevo nivel de importancia a los datos para la toma de decisiones y manejo de información relevante para su nivel operativo.



Big Data

la tendencia en el avance de la tecnología que ha abierto las puertas hacia un nuevo enfoque de entendimiento y toma de decisiones, la cual es utilizada para describir enormes cantidades de datos (estructurados, no estructurados y semi estructurados) que tomaría demasiado tiempo y sería muy costoso cargarlos a un base de datos relacional para su análisis. De tal manera que, el concepto de Big Data aplica para toda aquella información que no puede ser procesada o analizada utilizando procesos o herramientas tradicionales. Sin embargo, Big Data no se refiere a alguna cantidad en específico, ya que es usualmente utilizado cuando se habla en términos de petabytes y exabytes de datos.

Además del gran ***volumen*** de información, esta existe en una gran ***variedad*** de datos que pueden ser representados de diversas maneras en todo el mundo, por ejemplo de dispositivos móviles, audio, video, sistemas GPS, incontables sensores digitales en equipos industriales, automóviles, medidores eléctricos, veletas, anemómetros, etc., los cuales pueden medir y comunicar el posicionamiento, movimiento, vibración, temperatura, humedad y hasta los cambios químicos que sufre el aire, de tal forma que las aplicaciones que analizan estos datos requieren que la ***velocidad*** de respuesta sea lo demasiado rápida para lograr obtener la información correcta en el momento preciso. Estas son las características principales de una oportunidad para Big Data.



Teorema CAP

El teorema CAP, también llamado formalmente Teorema de Brewer, dice que un sistema de datos distribuido pude asegurar dos de estas tres propiedades: Consistencia, Disponibilidad y Tolerancia al particionado. Bien, que significa cada una:

La consistencia (Consistency), Todos los nodos deben ver los mismos datos al mismo tiempo, esto quiere decir que; cualquier cambio en los datos se debe aplicar en todos los nodos, y cuando se recupere el dato tiene que ser el mismo en todos los nodos. Esto se le llama consistencia atómica, y se consigue replicando la información en todos los nodos.

La disponibilidad (Availability), Cada petición en un nodo debe recibir y garantizar una confirmación si ha sido resuelta satisfactoriamente. En pocas palabras, se debe leer y escribir en todos los nodos.

La tolerancia al particionado (Partition Tolerance), El sistema debe funcionar a pesar de que haya sido dividido por un fallo de comunicación, garantizando la disponibilidad a pesar de que un nodo se separe del grupo sin importar la causa.

El teorema solo nos puede garantizar las siguientes combinaciones:

CP (Consistency & Partition): El sistema aplicara los cambios de forma consistente y aunque se pierda la comunicación entre nodos ocasionando el particionado, no se asegura que haya disponibilidad.

AP (Availability & Partition): El sistema siempre estará disponible a las peticiones, aunque se pierda la comunicación entre los nodos ocasionando el particionado, y en consecuencia por la pérdida de comunicación existirá inconsistencia porque no todos los nodos serán iguales.

CA (Consistency & Availability): El sistema siempre estará disponible respondiendo las peticiones y los datos procesados serán consistentes. En este caso no se puede permitir el particionado.

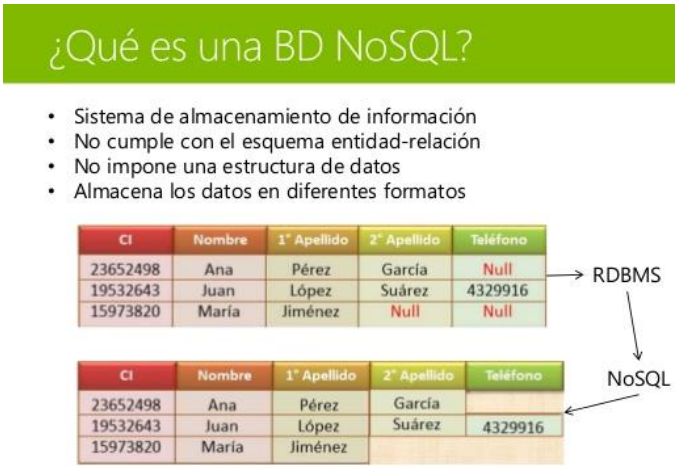
BD No SQL

Hablar de bases de datos NoSQL es hablar de estructuras que nos permiten almacenar información en aquellas situaciones en las que las bases de datos relacionales generan ciertos problemas debido principalmente a problemas de escalabilidad y rendimiento de las bases de datos relacionales donde se dan cita miles de usuarios concurrentes y con millones de consultas diarias.

Además de lo comentado anteriormente, las bases de datos No SQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad–relación. Tampoco utilizan una estructura de datos en forma de tabla donde se van almacenando los datos, sino que para el almacenamiento hacen uso de otros formatos como clave–valor, mapeo de columnas o grafos.

Esta forma de almacenar la información ofrece ciertas ventajas sobre los modelos relacionales. Entre las ventajas más significativas podemos destacar:

- Se ejecutan en máquinas con pocos recursos: Estos sistemas, a diferencia de los sistemas basados en SQL, no requieren de apenas computación, por lo que se pueden montar en máquinas de un coste más reducido.
- Escalabilidad horizontal: Para mejorar el rendimiento de estos sistemas simplemente se consigue añadiendo más nodos, con la única operación de indicar al sistema cuáles son los nodos que están disponibles.
- Pueden manejar gran cantidad de datos: Esto es debido a que utiliza una estructura distribuida, en muchos casos mediante tablas Hash.
- No genera cuellos de botella: El principal problema de los sistemas SQL es que necesitan transcribir cada sentencia para poder ser ejecutada, y cada sentencia compleja requiere además de un nivel de ejecución aún más complejo, lo que constituye un punto de entrada en común, que ante muchas peticiones puede ralentizar el sistema.



BD SQL

La sigla que se conoce como SQL corresponde a la expresión inglesa Structured Query Language (entendida en español como Lenguaje de Consulta Estructurado), la cual identifica a un tipo de lenguaje vinculado con la gestión de bases de datos de carácter relacional que permite la especificación de distintas clases de operaciones entre éstas. Gracias a la utilización del álgebra y de cálculos relacionales, el SQL brinda la posibilidad de realizar consultas con el objetivo de recuperar información de las bases de datos de manera sencilla.

El científico Edgar Frank Codd (1923–2003) fue quien propuso un modelo relacional para las bases de datos y creó un sublenguaje para acceder a los datos a partir del cálculo de predicados. En base al trabajo de Codd, IBM (International Business Machines) definió el lenguaje conocido como Structured English Query Language (SEQUEL).

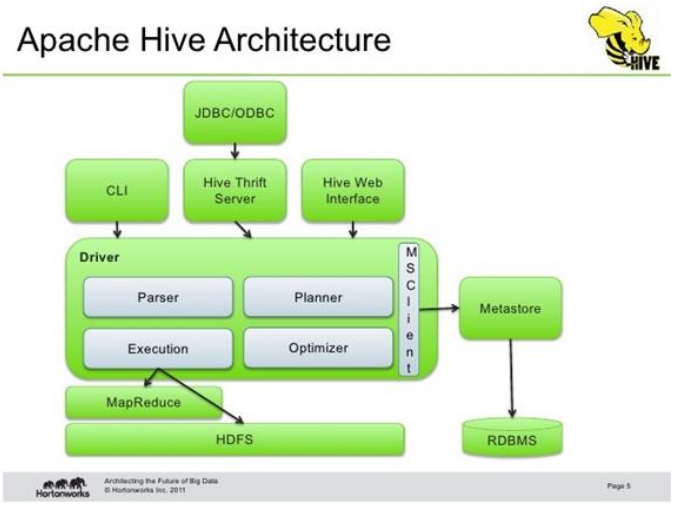
El SEQUEL se considera el antecesor de SQL, un lenguaje de cuarta generación que se estandarizó en 1986. La versión más primitiva de SQL, por lo tanto, fue la que se bautizó como SQL-86 (también conocida como SQL1).

En esencia, el SQL es un lenguaje declarativo de alto nivel ya que, al manejar conjuntos de registros y no registros individuales, ofrece una elevada productividad en la codificación y en la orientación a objetos. Una sentencia de SQL puede resultar equivalente a más de un programa que emplee un lenguaje de bajo nivel.

Una base de datos, dicen los expertos, implica la coexistencia de múltiples tipos de lenguajes. El denominado Data Definition Language (también conocido como DDL) es aquél que permite modificar la estructura de los objetos contemplados por la base de datos por medio de cuatro operaciones básicas. SQL, por su parte, es un lenguaje que permite manipular datos (Data Manipulation Language o DML) que contribuye a la gestión de las bases de datos a través de consultas.

Apache hive: Es el estándar por defecto para el manejo de consultas SQL interactivas sobre los datos manejados en hadoop. Ya que estos datos heterogéneos y no heterogéneos se almacenan secuencialmente como un esquema de lectura, hive se utiliza para poder consultar y usar la información procesable. Apache Hive soporta el análisis de grandes conjuntos de datos almacenados bajo HDFS de Hadoop y en sistemas compatibles como el sistema de archivos Amazon S3. Ofrece un lenguaje de consultas basado en SQL llamado HiveQL 5 con esquemas para leer y convertir consultas de forma transparente en MapReduce, Apache Tez6 y tareas Spark. Los tres motores de ejecución pueden correr bajo YARN. Para acelerar las consultas, Hive provee índices, que incluyen índices de bitmaps.7 Otras características de Hive incluyen:

- Es Familiar: Ya que consulta datos basado en lenguaje SQL
- Es Rápido: A pesar de manejar grandes conjuntos de datos, el tiempo de respuesta es muy aceptable
- Es escalable y extensible: A medida que se van agregando mas datos, y por consiguiente mas maquinas, estas se pueden adaptar sin sacrificar rendimiento.
- Es compatible: Funciona con herramientas tradicionales de integración de datos y análisis de datos.

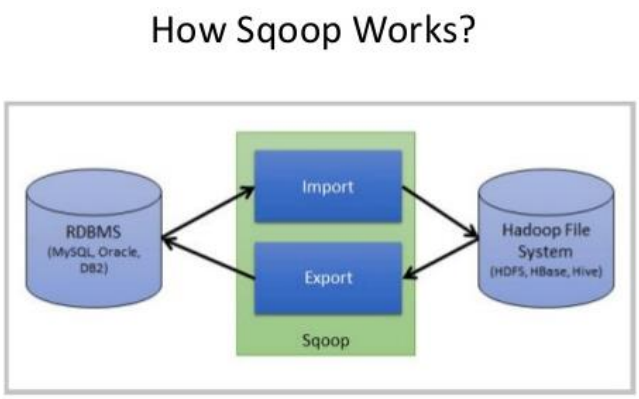


Apache Sqoop: Es el encargado de transferir datos entre hadoop y los almacenes de datos estructurados, haciendo una ejecución muy eficiente a un bajo costo, teniendo múltiple compatibilidad con muchas bases de datos relacionales. Con Sqoop, se pueden importar datos desde un sistema de base de datos relacional o un mainframe a HDFS. La entrada al proceso de importación es una tabla de base de datos o conjuntos de datos de mainframe. Para las bases de datos, Sqoop leerá la tabla fila por fila en HDFS. Para los conjuntos de datos de mainframe, Sqoop leerá los registros de cada conjunto de datos de mainframe en HDFS. El resultado de este proceso de importación es un conjunto de archivos que contiene una copia de la tabla o datasets importados. El proceso de importación se realiza en paralelo. Por esta razón, la salida estará en múltiples archivos. Estos archivos pueden ser archivos de texto delimitados (por ejemplo, con comas o tabulaciones que separan cada campo) o Avro o Sequence Files binarios que contienen datos de registro serializados.

Un subproducto del proceso de importación es una clase Java generada que puede encapsular una fila de la tabla importada. Esta clase es utilizada durante el proceso de importación por Sqoop. También se proporciona el código fuente de Java para esta clase, para su uso en el procesamiento posterior de MapReduce de los datos. Esta clase puede serializar y deserializar datos hacia y desde el formato SequenceFile. También puede analizar la forma de texto delimitado de un registro. Estas habilidades le permiten desarrollar rápidamente las aplicaciones de MapReduce que usan los registros almacenados en HDFS en su canal de procesamiento.
Después de manipular los registros importados (por ejemplo, con MapReduce o Hive) puede tener un conjunto de datos de resultados que luego puede exportar a la base de datos relacional. El proceso de exportación de Sqoop leerá un conjunto de archivos de texto delimitados de HDFS en paralelo, los analizará en registros y los insertará como nuevas filas en una tabla de base de datos de destino, para consumo de aplicaciones externas o usuarios.

Entre sus ventajas podemos encontrar:

- Importar datasets secuenciales desde mainframe: Satisface la creciente necesidad de mover datos de mainframe a HDFS
- Importación directa a ORCFiles: Compresión mejorada e indexación ligera para un rendimiento de consulta mejorado
- Importaciones de datos: Transfiere ciertos datos desde almacenamiento externo y EDW a Hadoop para optimizar la relación costo-efectividad del almacenamiento y procesamiento de datos combinados
- Transferencia de datos paralela: Para un rendimiento más rápido y una utilización óptima del sistema
- Copias de datos rápidas: Desde sistemas externos a Hadoop
- Análisis eficiente de datos: Mejora la eficiencia del análisis de datos combinando datos estructurados con datos no estructurados en un pool de datos de esquema en lectura
- Balanceo de carga: Mitiga el almacenamiento excesivo y el procesamiento de cargas a otros sistemas



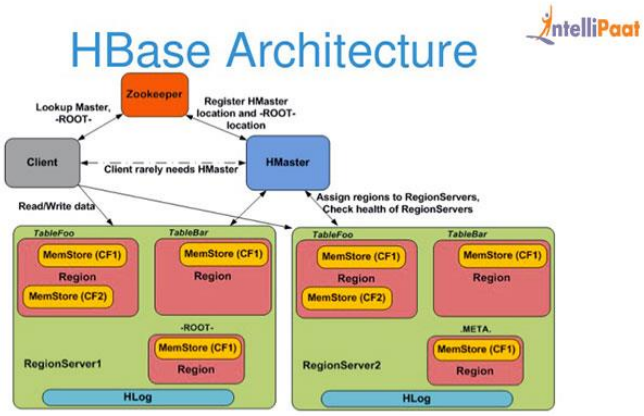
Apache HBase: Es una base de datos NoSql open source que permite el manejo de grandes conjuntos de datos, escalando linealmente para trabajar millones de filas y columnas, integrándose nativamente con hadoop y trabajando con otros motores de datos a través de YARN. Se usa en caso de tener tablas bastante grandes y datos dispersos o de estructura múltiple. Las tablas en HBase pueden servir como entrada o salida para tareas MapReduce en Hadoop, y se puede acceder a través del API en Java, como servicio REST, o con los API de conexión Avro y Thrift. Hbase es un almacén de datos orientado a columnas de tipo clave-valor y basado en Hadoop y HDFS. HBase se ejecuta sobre HDFS y es adecuado para acelerar operaciones de lectura y escritura en los grandes conjuntos de datos con un alto rendimiento y una baja latencia de entrada/salida.

HBase no reemplaza las bases de datos SQL clásicas, sin

embargo el proyecto Apache Phoenix proporciona una capa de SQL para Hbase así como acceso vía el API JDBC que puede ser integrado con diversas herramientas de analítica de datos e inteligencia de negocios. El proyecto Apache Trafodion proporciona un motor de consulta SQL con drivers ODBC y JDBC y protección de transacciones ACID distribuidas a través de múltiples consultas, tablas y filas utilizando HBase como motor de almacenamiento.

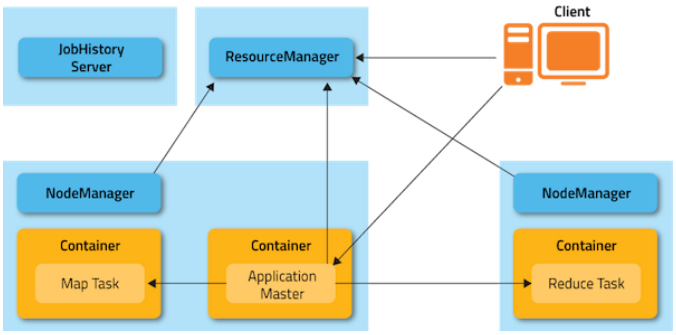
Sus características son:

- Tolerancia a fallos: Replicación en el centro de datos, operaciones atómicas y altamente consistentes a nivel de fila, alta disponibilidad a través de failover automático y equilibrado automático y balanceo de cargas de tablas
- Rapidez: Consultas casi en tiempo real, almacenamiento en memoria caché en memoria a través de la memoria caché de bloque y los filtros de floración, y procesamiento del lado del servidor a través de filtros y coprocesadores
- Es usable: El modelo de datos se adapta a una amplia gama de casos de uso, exportaciones de métricas a través de plugins File y Ganglia y API fácil de Java , así como API de Thrift y puerta de enlace REST



Apache YARN: Es el centro arquitectónico de Hadoop, el cual permite la interacción de varios motores de procesamiento de datos, ciencia de datos y manejo por lotes de estos, permitiendo un análisis más enfocado y simple. Se encarga de la seguridad y la administración de datos en los clústeres Hadoop además de proveer de almacenamiento y procesamiento lineal a las tecnologías previamente existentes en el centro de datos. YARN combina un administrador central de recursos que reconcilia la forma en que las aplicaciones utilizan los recursos del sistema de Hadoop con los agentes de administración de nodo que monitorean las operaciones de procesamiento de nodos individuales del clúster. Ejecutándose en clústeres de hardware básicos, Hadoop ha atraído un interés particular como zona de espera y de almacenamiento de datos para grandes volúmenes de datos estructurados y no estructurados destinados al uso en aplicaciones de analítica. Separar HDFS de MapReduce con YARN hace al ambiente Hadoop más adecuado para las aplicaciones operativas que no pueden esperar para que terminen los trabajos por lotes. Además de esto permite:

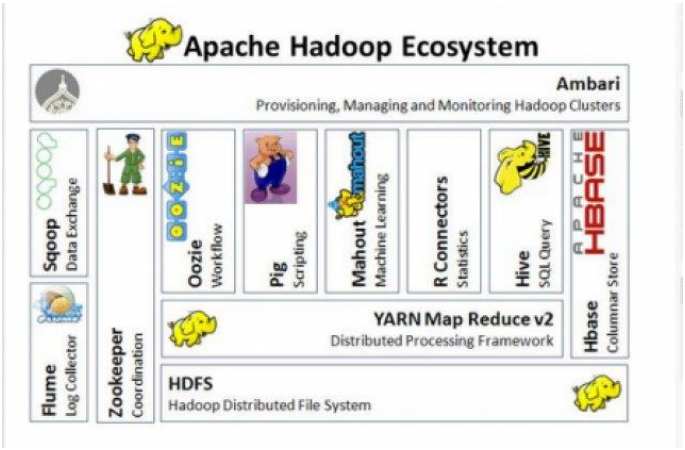
- Multi Tenant: YARN permite que los motores de acceso múltiple (ya sea de código abierto o de propietario) utilicen Hadoop como el estándar común para los motores por lotes, interactivos y en tiempo real que pueden acceder simultáneamente al mismo conjunto de datos, lo cual mejora el rendimiento de una empresa en sus inversiones con respecto a hadoop
- Utilización del clúster: La asignación dinámica de recursos de clúster de YARN mejora la utilización sobre reglas de MapReduce más estáticas utilizadas en las primeras versiones de Hadoop.
- Escalabilidad: El poder de procesamiento del centro de datos continúa expandiéndose rápidamente. El ResourceManager de YARN se centra exclusivamente en la programación y mantiene el ritmo a medida que los clústeres se expanden a miles de nodos que gestionan petabytes de datos.
- Compatibilidad: Las aplicaciones MapReduce existentes desarrolladas para Hadoop 1 pueden ejecutar YARN sin ninguna interrupción en los procesos existentes que ya funcionan



Apache Hadoop: Es una plataforma de software de código abierto para el almacenamiento distribuido y el procesamiento distribuido de conjuntos de datos muy grandes en clústeres de computadoras creados a partir de hardware básico. Los servicios de Hadoop proporcionan almacenamiento de datos, procesamiento de datos, acceso a datos, administración de datos, seguridad y operaciones.
Algunas razones para usar hadoop además de almacenar, administrar y analizar grandes cantidades de datos

estructurados y no estructurados de manera rápida, confiable, flexible y de bajo costo son:

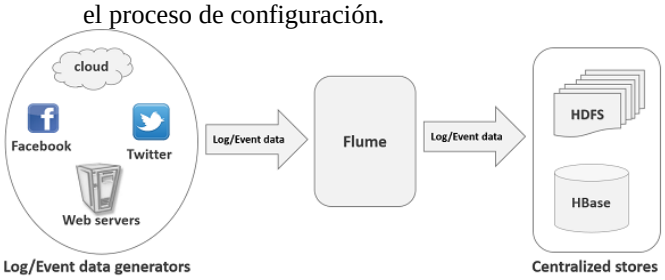
- Escalabilidad y rendimiento: el procesamiento distribuido de datos locales para cada nodo en un clúster permite a Hadoop almacenar, gestionar, procesar y analizar datos a escala de petabytes.
- Confiabilidad: los grandes clusters informáticos son propensos a la falla de los nodos individuales en el clúster. Hadoop es fundamentalmente resistente: cuando un nodo falla, el procesamiento se redirige a los nodos restantes en el clúster y los datos se vuelven a replicar automáticamente en preparación para futuras fallas de nodo.
- Flexibilidad: a diferencia de los sistemas de administración de bases de datos relacionales tradicionales, no es necesario crear esquemas estructurados antes de almacenar datos. Puede almacenar datos en cualquier formato, incluidos los formatos semiestructurados o no estructurados, y luego analizar y aplicar el esquema a los datos cuando se leen.
- Bajo costo: a diferencia del software propietario, Hadoop es de código abierto y funciona con hardware básico de bajo costo.



Apache Flume: Es un proyecto de nivel superior de Apache. Puede funcionar como un administrador de colas de eventos de propósito general, pero en el contexto de Hadoop se usa principalmente como un agregador de registros, recolectando datos de registros de diferentes fuentes y moviéndose a un almacén de datos centralizado.
Tiene una arquitectura sencilla y flexible basada en flujos de datos en streaming. Es robusto y tolerante a fallos, con mecanismos de fiabilidad configurables y muchos mecanismos de conmutación por error (failover) y recuperación. Utiliza un modelo de datos sencillo y extensible que permite la creación de aplicaciones analíticas en línea.

Un flujo de datos de Flume está compuesto por 5 componentes principales, los cuales son:

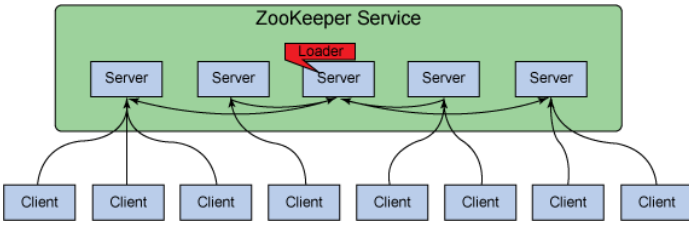
- **Eventos**: es la unidad básica de datos que es movida usando Flume. Es similar a un mensaje en JMS y es generalmente pequeño. Está compuesto por un header y un arreglo de bytes como cuerpo.
- **Fuentes**: recibe el evento de alguna entidad externa y lo guarda en un canal. La fuente debe de entender el tipo de evento que se le envia, por ejemplo, un evento Avro requiere una fuente Avro.
- **Canales**: es un almacenamiento interno pasivo con ciertas características específicas, por ejemplo, puede mover eventos muy rápido, pero no provee persistencia. Un canal basado en archivos si provee persistencia. Una fuente guarda un evento en el canal donde permanece hasta que es consumido por un pozo. Este almacenamiento temporal le permite a la fuente y al pozo correr de manera asíncrona.
- **Pozos**: remueve el evento del canal y lo pasa a un destino, como HDFS, o a otro agente. El pozo debe de tener como salida un evento ya apropiado para su destino.
- **Agentes**: es un contenedor para un flujo de datos de Flume. Es cualquier JVM física que esté ejecutando Flume. Un agente debe de contener por lo menos una fuente, canal y pozo, pero el mismo agente puede ejecutar múltiples fuentes, pozos y canales. Un camino particular para un flujo de datos es definido en el proceso de configuración.



Apache ZooKeeper: es un servidor open source que coordina procesos distribuidos de forma segura. ZooKeeper provee servicios operacionales para un clúster de Hadoop. Además provee un servicio de configuración distribuido, un servicio de sincronización y un registro de nombramiento para sistemas distribuidos. Las aplicaciones distribuidas usan ZooKeeper para almacenar y mediar actualizaciones para información de configuración importante.

ZooKeeper provee una muy simple interface y servicios, además trae los siguientes beneficios clave:

- **Rápido**: ZooKeeper es especialmente rápido con cargas de trabajo donde leer datos es más común que escribir. La proporción ideal de lectura/escritura es alrededor de 10:1
- **Seguro**: ZooKeeper está replicado sobre un set de hosts (llamados un conjunto) y los servidores están conscientes el uno del otro. Siempre que haya una masa crítica de servidores disponible, el servicio de ZooKeeper estará también disponible. No hay un solo punto de falla.
- **Simple**: Zookeeper mantiene un espacio de nombre jerárquico estándar, similar a los archivos y directorios.
- **Ordenado**: El servicio mantiene un registro de todas las transacciones, el cual puede ser usado para abstracciones de mayor nivel, como sincronización de primitivas.



Raspberry Pi: Es un computador de placa simple de bajo costo, creado con el objetivo de estimular las ciencias de la computación en las escuelas. Pero ya que es considerado un computador con un sistema operativo basado en linux, se puede trabajar como un nodo esclavo de un ecosistema big data, habiendo instalado previamente todas las herramientas que se van a usar en el ecosistema (hadoop, yarn, sqoop, hive, hbase, etc). Ofreciendo así una solución de bajo costo para poder crear un clúster funcional basado en Big Data.

Para que funcione, basta con que añadamos nosotros mismos un medio de almacenamiento (como por ejemplo una tarjeta de memoria SD), enchufarlo a la corriente gracias a cualquier cargador de tipo microUSB (el mismo que sirve para recargar la mayoría de los teléfonos móviles, cuyo coste es ínfimo) y, si lo deseamos, incorporar un chasis para que todo quede a buen recaudo y su apariencia sea más estética. Estos pueden ser desde cajas predeterminadas hasta una que fabriquemos nosotros mismos echándole grandes dosis de imaginación.

La fundación de Raspberry Pi pone a disposición desde su página web Raspbian, una distribución de Linux basada en Debian, pero también podemos recurrir a muchas de las distribuciones específicas que la comunidad de usuarios ha desarrollado para diversos fines.

En función del modelo que escojamos, dispondremos de más o menos opciones de conexión, pero siempre tendremos al menos un puerto de salida de video HDMI y otro de tipo RCA, minijack de audio y un puerto USB 2.0 al que conectar un teclado y ratón.

En cuanto a conexión de red se refiere, podemos disponer de Ethernet para enchufar un cable RJ-45 directamente al router o recurrir a adaptadores inalámbricos WiFi. En este caso, eso sí, conviene que nos decantemos por la Raspberry Pi que incorpora dos puertos USB, ya que si no no podremos enchufar el teclado y el ratón.

VIII. DESCRIPCION DEL PROCESO

Hadoop tiene un gran potencial y es uno de los proyectos más conocidos para big data. A continuación, instalaremos y configuraremos un clúster Hadoop usando Raspberrys Pi. Nuestro clúster consistirá en tres nodos (un maestro y dos esclavos).

Como las raspberrys tienen un precio bajo y Hadoop es de código abierto, ofrecen una gran oportunidad para cualquiera que quiera comenzar a trabajar con big data. Configuraremos el clúster desde una estación de trabajo Linux, trabajaremos exclusivamente con comandos de Linux. El clúster se puede construir con otras familias de sistemas operativos, y los pasos generales de este tutorial seguirán siendo útiles, pero los comandos específicos serían diferentes.

IX. SISTEMA OPERATIVO EMPLEADO

Raspbian: Raspbian es una distribución del sistema operativo GNU/Linux y por lo tanto libre basado en Debian 8 para la Raspberry Pi, orientado a la enseñanza de informática. Técnicamente el sistema operativo es un port no oficial de Debian armhf para el procesador de Raspberry Pi, con soporte optimizado para cálculos en coma flotante por hardware.

La distribución usa LXDE como escritorio y Midori como navegador web. Además, contiene herramientas de desarrollo como IDLE para el lenguaje de programación Python o Scratch, y diferentes ejemplos de juegos usando los módulos Pygame.

Destaca también el menú "raspi-config" que permite configurar el sistema operativo sin tener que modificar archivos de configuración manualmente. Entre sus funciones, permite expandir la partición root para que ocupe toda la tarjeta de memoria, configurar el teclado, aplicar overclock, etc.

X. CONFIGURACION DE DISPOSITIVOS Y PROCEDIMIENTO

Descarga e instalación del sistema operativo en las tarjetas micro SD:
Cualquier SO que sea compatible con Hadoop funcionará. En nuestro caso, seleccionamos Raspbian Jessy Lite como nuestro sistema operativo. Es un sistema operativo liviano, diseñado específicamente para raspberrys. Se puede descargar desde el sitio web oficial de la Fundación Raspberry Pi: https://www.raspberrypi.org/downloads/raspbian/
Una vez que descargado nuestro archivo .img, podemos instalarlo en una tarjeta micro SD. Este proceso sobrescribirá cualquier dato que pueda tener la tarjeta SD, por lo que se recomienda usar una tarjeta vacía.
Hay muchas maneras de instalar el sistema operativo en la tarjeta SD. Usaremos el comando dd Linux:
sudo dd if=2016-02-26-raspbian-jessie-lite.img of=/dev/sdd

El parámetro **if** es el archivo de imagen que hemos descargado. El parámetro **of** es el punto de montaje de nuestra tarjeta micro SD.
Repetiremos esta instalación en cada tarjeta micro SD.
Una vez hecho esto, insertamos las tarjetas micro SD en las Raspberries e iniciamos sesión en ellas a través de SSH como el usuario predeterminado "pi" con la contraseña " raspberry".
Es posible que necesite expandir el sistema de archivos para usar la capacidad total de las tarjetas micro SD. Si este es el caso, inicie sesión en las Raspberries con SSH y ejecute:
sudo raspi-config

A continuación, seleccionamos "Expandir sistema de archivos". Una vez hecho esto, reiniciamos la Raspberry, así:
shutdown -r now

Crear la instalación y los directorios de HDFS:
Necesitamos un directorio de instalación para Hadoop. También necesitamos un directorio donde el HDFS (Hadoop Distributed File System) colocará sus archivos. En nuestro caso, seleccionamos /opt/hadoop/ como el directorio de instalación y /opt/hadoop_tmp/ como la ruta que usará HDFS. Así que creemos esos directorios
sudo mkdir /opt/hadoop/
sudo mkdir /opt/hadoop_tmp/

De nuevo, nos aseguramos de hacer esto en todas las Raspberry. Y tomamos nota del directorio HDFS, lo necesitaremos más adelante mientras editamos los archivos de configuración.

Instalacion de Java:
Como Hadoop lo requiere, Java debe instalarse en todos los nodos. Hadoop 2.7 y versiones posteriores requieren Java 7. Hadoop 2.6 aún es compatible con Java 6. Instalamos la versión compatible con nuestro Hadoop deseado. En Raspbian, para instalar Oracle Java 8 hacemos esto:
sudo apt-get update
sudo apt-get install oracle-java8-jdk

Y verificamos que se haya instalado correctamente:
java -version
java version "1.8.0_65"
Java(TM) SE Runtime Environment (build 1.8.0_65-b17)
Java HotSpot(TM) Client VM (build 25.65-b01, mixed mode)

Configuracion de la conectividad:
Lo que necesitamos es que cada Raspberry tenga su propia dirección IP y que puedan conectarse entre sí sin problemas.
Para facilitar nuestro trabajo, a partir de ahora utilizaremos un alias para cada nodo. Hacemos una lista de todas las direcciones IP y las agregamos al archivo /etc/hosts en cada Raspberry. En nuestro caso, se veria así:
x.xx.xxx.xxx master
x.xx.xxx.xxx slave-01
x.xx.xxx.xxx slave-02

Generación y replicación de claves SSH:
Necesitamos todas las Raspberries para poder comunicarnos entre nosotros sin pedir contraseñas. Para eso, generaremos claves SSH en todas ellas:
ssh-keygen -t rsa -b 4096 -C your_email@example.com

Y luego replicamos las llaves de todas las Raspberries
ssh-copy-id user@master
ssh-copy-id user@slave-01
ssh-copy-id user@slave-02

Después de que hayamos terminado, cada Raspberry debe tener todas las claves. Este es el paso donde la mayoría de las personas comete errores, y la mayoría de los problemas al iniciar el clúster provienen de una configuración errónea de las claves SSH. Así que verificamos que todos los nodos se puedan conectar entre sí.

Agregar variables de entorno:
Ahora, agregaremos algunas variables de entorno de Hadoop a nuestro archivo ~/.bashrc.
-- HADOOP ENVIRONMENT VARIABLES START --
export HADOOP_HOME=/opt/hadoop/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-

Djava.library.path=$HADOOP_HOME/lib"
# -- HADOOP ENVIRONMENT VARIABLES END -- #
export JAVA_HOME=/usr/lib/jvm/jdk-8-oracle-arm32-vfp-hflt/jre

Instalacion de Hadoop en el namenode:
El namenode es nuestro nodo maestro. Los nodos esclavos se denominan como nodos de datos. Instalaremos Hadoop en nuestro namenode y realizaremos algunas configuraciones, luego, trabajaremos en los nodos de datos. Como esta es la configuración básica, este y los siguientes pasos se realizarán solo en el namenode. Luego copiaremos esta configuración básica a los 2 nodos restantes, y luego realizaremos personalizaciones específicas.
Vamos al directorio de instalación que creamos en el segundo paso. Seleccionamos una versión de Hadoop desde la página de descarga y obtenemos la URL de la barra de navegacion. Los siguientes comandos descargarán un paquete de Hadoop y lo descomprimirán:
cd /opt/hadoop/
sudo wget http://www-us.apache.org/dist/hadoop/common/hadoop-2.6.4/hadoop-2.6.4.tar.gz
sudo tar xvf hadoop-2.6.4.tar.gz
mv hadoop-2.6.4 hadoop

Agregar los archivos del maestro y los esclavos:
Este paso solo debe hacerse en el nodo maestro. Debemos agregar dos archivos en $HADOOP_HOME/hadoop/etc/hadoop/, uno de ellos se llamará **master** y el otro se llamará **slaves**.
En este nuevo archivo **master**, solo necesitamos agregar el nombre de host o el alias del nodo maestro. Entonces, en nuestro caso, este archivo solo contendrá una línea:
master

En el archivo de **slaves** solo agregaremos nuestros nodos de esclavos, un alias por línea. En nuestro caso:
slave-01
slave-02

Configurar HDFS:
Es necesario crear un par de directorios en nuestro directorio HDFS. En el namenode. vamos al directorio HDFS (/opt/hadoop_tmp/ en nuestro caso) y creeamos un directorio llamado "hdfs", luego, dentro de ese directorio creeamos otro directorio llamado "namenode".
sudo mkdir /opt/hadoop_tmp/hdfs
sudo mkdir /opt/hadoop_tmp/hdfs/namenode

A continuación, iremos a cada nodo esclavo y haremos lo mismo, excepto que el directorio más nuevo se llamará " datanode":
sudo mkdir /opt/hadoop_tmp/hdfs
sudo mkdir /opt/hadoop_tmp/hdfs/datanode

Copiar la configuración básica a los nodos esclavos:
Los pasos anteriores para la instalación de Hadoop se realizaron solo en el nodo maestro. Ahora tomaremos esos archivos y los copiaremos a las Raspberries restantes.
En el nodo maestro, ejecutamos los siguientes comandos para copiar la configuración básica a los esclavos (también se puede hacer con scp):
rsync -avxP <$HADOOP_HOME> <hadoop_user>@<slave-hostname>:<$HADOOP_HOME>

Y en nuestro caso, quedaría asi:
rsync -avxP /opt/hadoop/ user@slave-01:/opt/hadoop/
rsync -avxP /opt/hadoop/ user@slave-02:/opt/hadoop/

Editar los archivos de configuración:
Hay varios archivos XML que deben modificarse de acuerdo con nuestra arquitectura. Estos cambios deben realizarse en el maestro y en todos los nodos esclavos.
Actualizamos core-site.xml (ubicado en $ HADOOP_HOME/hadoop/etc/hadoop/core-site.xml), cambiaremos "localhost" a nuestro nombre de host del nodo principal, IP o alias. En nuestro caso, sería así:

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://master:9000/</value>
  </property>
  <property>
    <name>fs.default.FS</name>
    <value>hdfs://master:9000/</value>
  </property>
</configuration>
```

Actualizamos tambien hdfs-site.xml (ubicado en $ HADOOP_HOME/hadoop/etc/hadoop/hdfs-site.xml), cambiaremos el factor de replicación por el número de nodos de datos que tengamos, y especificaremos los directorios de nodo de datos y namenodes (que creamos en el paso anterior), así como agregar una dirección http:

```
<configuration>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>/opt/hadoop_tmp/hdfs/datanode</value>
    <final>true</final>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>/opt/hadoop_tmp/hdfs/namenode</value>
    <final>true</final>
```

```
  </property>
  <property>
    <name>dfs.namenode.http-address</name>
    <value>master:50070</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>11</value>
  </property>
</configuration>
```

Actualizamos yarn-site.xml (ubicado en $ HADOOP_HOME/hadoop/etc/hadoop/yarn-site.xml). Hay 3 propiedades que necesitamos actualizar con nuestro nombre de host o alias de nodo maestro:

```
<configuration>
  <property>
    <name>yarn.resourcemanager.resource-
tracker.address</name>
    <value>master:8025</value>
  </property>
  <property>
    <name>yarn.resourcemanager.scheduler.address</name>
    <value>master:8035</value>
  </property>
  <property>
    <name>yarn.resourcemanager.address</name>
    <value>master:8050</value>
  </property>
</configuration>
```

Actualizamos por ultimo mapred-site.xml (ubicado en $ HADOOP_HOME/hadoop/etc/hadoop/mapred-site.xml), agregaremos las siguientes propiedades:

```
<configuration>
  <property>
    <name>mapreduce.job.tracker</name>
    <value>master:5431</value>
  </property>
  <property>
    <name>mapred.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

Formatear el namenode y comenzar los servicios:
Vamos al namenode, en $HADOOP_HOME/bin/ ejecutamos lo siguiente:

```
./hdfs namenode -format
```

Eso formateará el maestro como un namenode adecuado. Finalmente, comenzamos los servicios. Anteriormente, Hadoop utilizaba el script start-all.sh para esto, pero ha quedado obsoleto y el método recomendado ahora es utilizar los scripts start-<service>.sh de forma individual. En cada nodo, vamos a $HADOOP_HOME/sbin/ y ejecutamos lo siguiente:

```
./start-yarn.sh
./start-dfs.sh
```

XI. CONCLUSIONES

- Es posible realizar un clúster de hadoop con raspberry pi de forma rápida haciendo uso de una guía como esta
- Se puede simular un entorno Big Data a un bajo costo haciendo uso de un clúster con raspberrys pi
- Es importante entender todos los componentes de un ecosistema Big Data antes de implementar cualquiera de ellos

XII. BIBLIOGRAFIA

1. http://alejandria.poligran.edu.co/jspui/bitstream/10823/683/2/Trabajo_De_Grado-Big_Tracking.%20V%204.0.pdf
2. https://www.iit.comillas.edu/pfc/resumenes/555ce109bd5fe.pdf
3. https://dataiq.com.ar/blog/por-que-es-importante-big-data/
4. https://www.dnp.gov.co/Paginas/Big-Data-Colombia-entra-en-la-revoluci%C3%B3n-de-los-datos-.aspx
5. https://www.raspberrypi.org/help/faqs/
6. https://elinux.org/RPi_raspi-config
7. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-015-0014-3
8. http://searchcio.techtarget.com/opinion/Ten-big-data-case-studies-in-a-nutshell
9. https://www.idgenterprise.com/resource/marketing-tools/big-data-and-analytics-the-big-picture-infographic/
10. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-015-0032-1
11. http://ieeexplore.ieee.org/abstract/document/7055837/?reload=true
12. https://storm.apache.org/
13. https://spark.apache.org/mllib/
14. http://mahout.apache.org/
15. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-015-0030-3
16. http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/printable_report.pdf
17. https://books.google.com.co/books?hl=es&lr=&id=kYC3YCyl_tkC&oi=fnd&pg=PR5&ots=qj62FJdhWD&sig=r2NI9c6irwiH4uE7EDLKjrtO6zI&redir_esc=y#v=onepage&q&f=false

18. https://dl.acm.org/citation.cfm?id=1015408
19. http://ieeexplore.ieee.org/document/1232271/
20. https://dl.acm.org/citation.cfm?doid=2168931.2168943
21. https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf
22. https://datafloq.com/read/3vs-sufficient-describe-big-data/166
23. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-014-0008-6
24. http://online.liebertpub.com/doi/abs/10.1089/big.2013.0006
25. https://dl.acm.org/citation.cfm?doid=1327452.1327492
26. https://dl.acm.org/citation.cfm?id=2523633
27. https://dl.acm.org/citation.cfm?doid=2094114.2094118
28. https://dl.acm.org/citation.cfm?id=1376726
29. https://journalofbigdata.springeropen.com/articles/10.1186/s40537-016-0056-1
30. http://iopscience.iop.org/article/10.1088/1742-6596/664/5/052023/meta
31. https://kops.uni-konstanz.de/handle/123456789/28647
32. https://dl.acm.org/citation.cfm?id=2228301 https://developer.ibm.com/recipes/tutorials/building-a-hadoop-cluster-with-raspberry-pi/
33. https://definicion.de/sql/
34. https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf
35. https://www.swapbytes.com/teorema-cap-base-datos/
36. https://www.ibm.com/developerworks/ssa/local/im/que-es-big-data/
37. https://www.cs.us.es/cursos/bd-2014/IntroDataScience.pdf
38. http://fc5scrim.blogspot.com.co/2015/08/los-cientifico-de-datos.html
39. https://www.revistagerentepyme.com/de-la-data-al-big-data-innovacion-para-entender-al-consumidor/
40. https://www.genbetadev.com/bases-de-datos/nosql-clasificacion-de-las-bases-de-datos-segun-el-teorema-cap
41. https://es.slideshare.net/geeklon/no-sql-bases-de-datos-no-relacionales (4/16)
42. https://es.123rf.com/photo_54027643_sql-lenguaje-de-consulta-estructurado-ilustraci%C3%B3n-s%C3%ADmbolo-de-base-de-datos-vectoriales-concepto-plana.html
43. https://es.wikipedia.org/wiki/Apache_Hive
44. https://intellipaat.com/blog/what-is-apache-hive/
45. https://sqoop.apache.org/docs/1.4.6/SqoopUserGuide.html#_introduction
46. https://www.slideshare.net/pavan5780/apache-sqoop-72298037
47. https://thirdeyedata.io/apache-hbase/architecture-of-apache-hbase/
48. http://searchdatacenter.techtarget.com/es/definicion/Apache-Hadoop-YARN-Yet-Another-Resource-Negotiator
49. http://blog.cloudera.com/blog/2014/05/how-apache-hadoop-yarn-ha-works/
50. http://www.dataprix.com/software-empresa/apache/apache-hadoop
51. **https://www.ibm.com/developerworks/library/bd-zookeeper/**
52. http://computerhoy.com/noticias/hardware/que-es-raspberry-pi-donde-comprarla-como-usarla-8614
53. https://www.rs-online.com/designspark/raspberry-pi
54. https://bigdatadummy.com/2017/02/07/apache-flume/